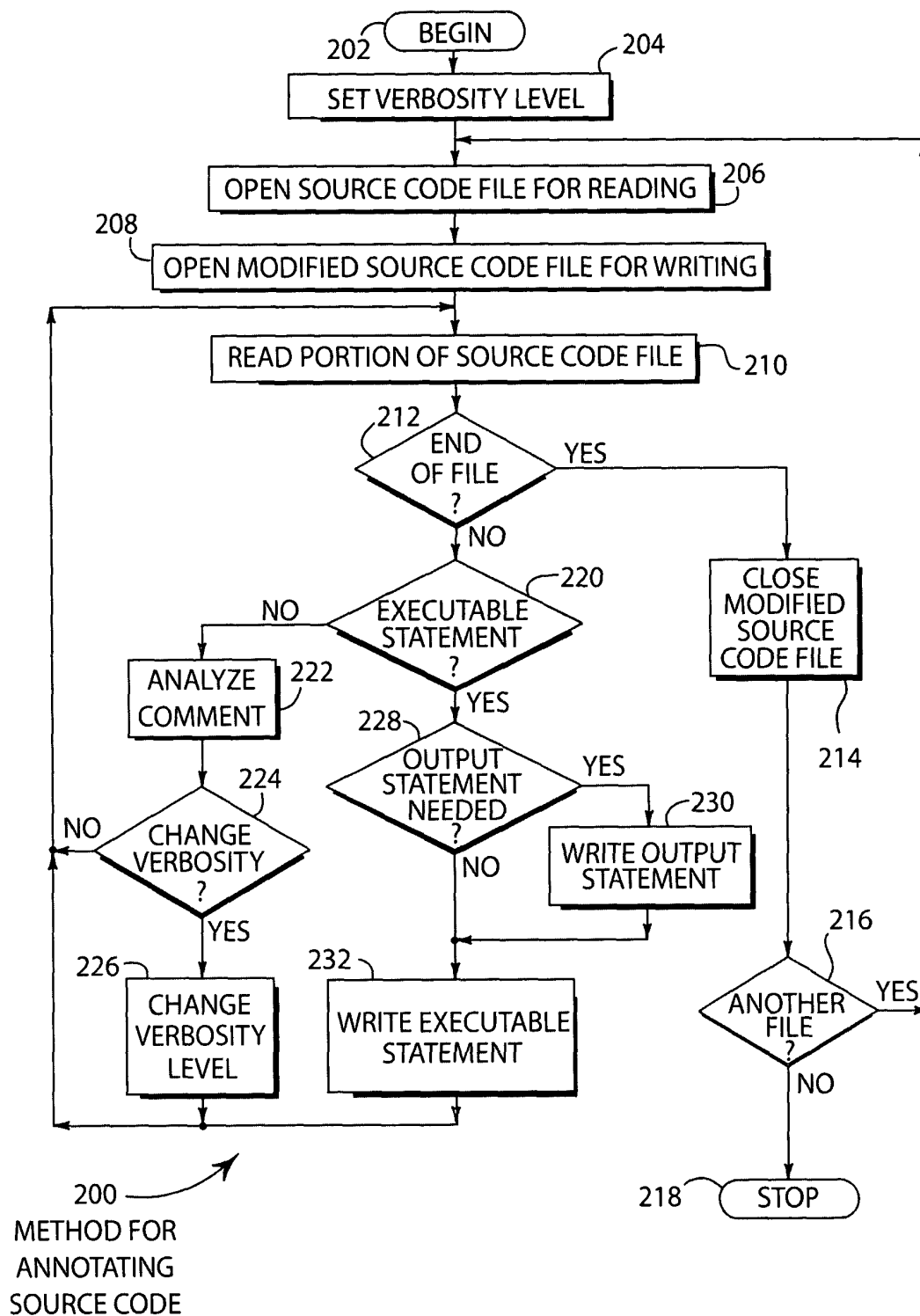


FIGURE 1

**FIGURE 2**

300
SOURCE CODE

```

int main( int argc, char *argv[] ) {
    int inside_comment = 0, trace_declared = 0, trace_opened = 0;
    int i, n;
    char file_name[MAX_LINE_LEN];
    FILE *infile, *outfile;
    char save_dir[MAX_LINE_LEN];

    if (!(argc == 4)) {
        printf("Usage: codetracer <input path> <output path> <file spec>\n\ncodetracer
        testdir outdir *.cpp\n");
        printf("or\n\n\ncodetracer this outdir *.cpp\n");
        exit( 0 );
    }
    else {
        printf( "Program codetracer version %s\n", VERSION );
    }
    // save path of the directory where we start
    if ( _getcwd( save_dir, MAX_PATH ) == NULL ) {
        printf( "Could not find path of current directory\n" );
        exit( -2 );
    }
    ..etc.

```

FIGURE 3

```

FILE *trace;

// <<<<< This line is inserted by the tool

int main( int argc, char *argv[] ) {
    trace = fopen( "trace.txt", "wt" ); // <<<<< This line is inserted by the tool
    fprintf( trace, "codetracer.cpp 0028: int main( int argc, char *argv[] ) {\n" );
    fprintf( trace, "codetracer.cpp 0029: int inside_comment = 0, trace_declared = 0,
    trace_opened = 0;\n" );
    int inside_comment = 0, trace_declared = 0, trace_opened = 0;
    fprintf( trace, "codetracer.cpp 0030: int i, n;\n" );
    int i, n;
    fprintf( trace, "codetracer.cpp 0031: char file_name[MAX_LINE_LEN];\n" );
    char file_name[MAX_LINE_LEN];
    fprintf( trace, "codetracer.cpp 0032: FILE *infile, *outfile;\n" );
    FILE *infile, *outfile;
    fprintf( trace, "codetracer.cpp 0033: char save_dir[MAX_LINE_LEN];\n" );
    char save_dir[MAX_LINE_LEN];

    fprintf( trace, "codetracer.cpp 0035: if ( ! ( argc == 4 ) ) {\n" );
    if ( ! ( argc == 4 ) ) {
        printf( "Usage: codetracer <input path> <output path> <file spec>\n\ncodetracer
        testdir outdir *.cpp\n" );
        printf( "or\n\n\ncodetracer this outdir *.cpp\n" );
        exit( 0 );
    }
    fprintf( trace, "codetracer.cpp 0038: exit( 0 );\n" );
    exit( 0 );
}
else {
    fprintf( trace, "codetracer.cpp 0040: else {\n" );
    printf( "Program codetracer version %s\n", VERSION );
}
fprintf( trace, "codetracer.cpp 0043: // save path of the directory where we start\n" );
// save path of the directory where we start
fprintf( trace, "codetracer.cpp 0044: if ( _getcwd( save_dir, _MAX_PATH ) == NULL )
{\n" );
if ( _getcwd( save_dir, _MAX_PATH ) == NULL ) {
    printf( "Could not find path of current directory\n" );
    exit( -2 );
}
fprintf( trace, "codetracer.cpp 0046: exit( -2 );\n" );
exit( -2 );
}
}
..etc.

```

400
 ANNOTATED
 SOURCE CODE

FIGURE 4

```

502 FILENAME 504 LINE NUMBER
codetracer.cpp 0028: int main( int argc, char *argv[] ) {
codetracer.cpp 0030:     int inside_comment = 0, trace_declared = 0, trace_opened = 0;
codetracer.cpp 0031:     int i, n;
codetracer.cpp 0032:     char file_name[MAX_LINE_LEN];
codetracer.cpp 0033:     FILE *infile, *outfile;
codetracer.cpp 0034:     char save_dir[MAX_LINE_LEN];
codetracer.cpp 0036:     if( !(argc == 4) ) {
codetracer.cpp 0041:     else {
codetracer.cpp 0045:         // save path of the directory where we start
codetracer.cpp 0046:         if ( _getcwd( save_dir, MAX_PATH ) == NULL ) {
codetracer.cpp 0051:         if( !strcmp( argv[1], "this" ) ) {
codetracer.cpp 0064:         else {
codetracer.cpp 0065:             n = FindFiles( argv[3] );

utility.cpp 0118: static int tree_depth = 0, i = 0;
utility.cpp 0119: struct finddata_t file;
utility.cpp 0120: long hFile;
utility.cpp 0121: int success = 1;
utility.cpp 0122: static int index = 0;
utility.cpp 0124: // search for the first file in the current directory
utility.cpp 0125: if( (hFile = _findfirst( spec1, &file )) == -1L ) {
utility.cpp 0130: while( success != -1L ) {
utility.cpp 0131:     if( file.name[0] != '.' ) {
utility.cpp 0132:         if( file.attrib & _A_SUBDIR ) == _A_SUBDIR ) {
utility.cpp 0144:         else { // it is a real filename
utility.cpp 0145:             strcpy( file_name_array[index], file.name );
utility.cpp 0146:             index++;
utility.cpp 0150:         // read next file and see if success
utility.cpp 0151:         success = _findnext( hFile, &file );
utility.cpp 0131:         if( file.name[0] != '.' ) {
utility.cpp 0132:             if( file.attrib & _A_SUBDIR ) == _A_SUBDIR ) {
utility.cpp 0144:             else { // it is a real filename
utility.cpp 0145:                 strcpy( file_name_array[index], file.name );
utility.cpp 0146:                 index++;
utility.cpp 0150:         // read next file and see if success
utility.cpp 0151:         success = _findnext( hFile, &file );
utility.cpp 0154:         if( tree_depth > 0 ) { // go up in the directory tree
utility.cpp 0158:         return index;

codetracer.cpp 0067: max_name_length = 0;
codetracer.cpp 0068: for( i = 0; i < n; i++ ) {
codetracer.cpp 0069:     if( (int)max_name_length < (int)strlen( file_name_array[i] ) ) {

```

OUTPUT
FILE
500

FIGURE 5